Ministério da Educação
Fundação Universidade Federal de Mato Grosso do Sul
Faculdade de Computação - Laboratório de Banco de Dados
Prof. Dr. Marcio Silva

Descrição do Trabalho Prático
Em grupo (máximo de 3 integrantes)
Entrega (22/06/2024, 23h59 - Horário do MS)

Informe os temas escolhidos aqui: https://forms.gle/6Jpv6ozjH9XBdaEg9
Fórum no AVA para quem está sem GRUPO
https://ava.ufms.br/mod/forum/view.php?f=112642

1 Introdução

O Trabalho Prático (TR) da disciplina de Laboratório de Banco de Dados consiste na modelagem, projeto e construção de um banco de dados relacional. Além disso, seu trabalho deve estar relacionado a um dos temas propostos a seguir. Este trabalho exigirá pesquisa adicional sobre SQL e o minimundo proposto para solucionar problemas reais apresentados nos requisitos. Será necessário a escolha de uma linguagem de programação para a interação com o banco de dados, eu recomendo utilizar Python, mas também poderá ser utilizado JAVA, PHP, Javascript ou Typescript.

Neste trabalho não é necessário nenhuma técnica de inteligência artificial, apenas consultas SQL ou heurísticas próprias, implementadas na linguagem escolhida, para tentar solucionar os problemas propostos. Nenhuma interface gráfica ou WEB será necessária neste trabalho, apenas interações via linha de comando.

2 Temas

Cada grupo deve escolher apenas um dos temas a seguir para desenvolver o seu projeto. As entidades ou tabelas propostas são apenas um ponto de partida e novas tabelas devem propostas para atender os requisitos e este será um ponto a ser avaliado em cada projeto entregue. **Após a escolha do tema, este não poderá ser trocado em hipótese alguma.**

2.1 TEMA 1: Sistema de Detecção de Anomalias em Transações Bancárias

Desenvolvimento de um sistema relacional capaz de armazenar, consultar e analisar transações financeiras de clientes de uma instituição bancária, com foco em detecção de comportamentos anômalos que podem indicar fraudes, erros ou uso indevido.

2.1.1 Objetivo Geral

Projetar e implementar uma solução completa que una modelagem de dados robusta, uso intensivo de SQL avançado (incluindo janelas, subqueries, agregações complexas), e lógica de negócios com programação para detectar transações bancárias atípicas com base em padrões históricos e regras heurísticas.

2.1.2 Escopo e Requisitos

Criar um modelo relacional normalizado, contendo ao menos as entidades `clientes, contas, transacoes, dispositivos, transacoes e alertas_anomalia`. Considerar integridade referencial, uso de chaves primárias e estrangeiras, e índices apropriados. Outros relacionamentos e tabelas auxiliares podem ser incluídos conforme a proposta do grupo.

Regras de Detecção de Anomalias

Deverão ser implementadas pelo menos 5 regras heurísticas, como por exemplo:

- Transações acima de um valor específico, fora do padrão histórico do cliente.
- Transações em localizações geográficas muito distantes em curto intervalo de tempo (ex: Rio de Janeiro e Londres em 10 min).
- Transações em horário incomum (ex: entre 2h e 5h) com valor elevado.
- Uso de um novo dispositivo para um cliente, sem histórico.
- Variação súbita no padrão de valores ou frequência de transações em comparação com os últimos 30 dias.

Linguagem de Programação

- Utilizar uma linguagem de alto nível (ex: Python, JavaScript/Node.js, Java) para:
  - Simular ou importar um conjunto consistente de dados (Mil transações).
  - Executar rotinas de análise e detecção de anomalias via conexão com o banco (via ORM ou queries diretas).
  - Gerar e registrar os alertas de anomalia.

Relatórios

Gerar relatórios como:

- Top 10 clientes com mais transações suspeitas.
- Lista de horário de maior incidência de alertas e quais são os tipos de alertas.
- Evolução temporal do número de alertas.

2.2 TEMA 2: Sistema de Recomendação de Cardápios Saudáveis com Restrições Nutricionais

Desenvolvimento de um sistema relacional capaz de armazenar informações nutricionais de alimentos, perfis de usuários com restrições alimentares e preferências pessoais, e gerar recomendações automatizadas de cardápios saudáveis, respeitando necessidades nutricionais e condições de saúde específicas (como diabetes, hipertensão, intolerância à lactose, etc.).

2.2.1 Objetivo Geral

Projetar e implementar um sistema de recomendação de cardápios personalizados, com base em um modelo de banco de dados relacional normalizado e uso intensivo de consultas SQL para combinar alimentos, refeições e perfis de usuário. A solução deverá oferecer recomendações baseadas em regras nutricionais e restrições definidas individualmente.

2.2.2 Escopo e Requisitos

Modelagem

O sistema deverá contemplar pelo menos as seguintes entidades:

- *usuarios* (informações pessoais dos usuários do sistema. Projete atribtos que facilitem a solução do problema)
- *alimentos* (nome, grupo alimentar, macronutrientes, calorias, sódio, entre outras informações que julgarem necessárias.)
- *refeicoes* (conjuntos de alimentos, horário sugerido, tipo da refeição: café, almoço, etc.)
- *restricoes* (diabetes, hipertensão, alergias, vegetarianismo, etc.)
- *recomendacoes* (registro de cardápios sugeridos para cada usuário)
- *avaliacoes* (feedback do usuário sobre os cardápios)

Outros relacionamentos e tabelas auxiliares podem ser incluídos conforme a proposta do grupo.

Regras de Geração de Cardápio

Implementar pelo menos 5 regras nutricionais, como por exemplo:

- Limitar ingestão de sódio para usuários hipertensos.
- Limitar índice glicêmico para diabéticos.
- Cardápios com restrição de lactose ou glúten para intolerantes.
- Respeitar meta calórica diária e proporções de macronutrientes (proteína, carboidrato, gordura).
- Combinar alimentos variados ao longo do dia para melhorar adesão e diversidade nutricional.

O grupo deve propor outras regras de geração de cardápio, não se limitando às citadas anteriormente.

Linguagem de Programação

Utilizar uma linguagem para:

- Cadastrar usuários e suas preferências/restrições.
- Consultar o banco e gerar sugestões de cardápios personalizados com base em consultas SQL.
- Registrar avaliações e histórico de sugestões.

É preciso cadastrar pelo menos 100 alimentos, 20 usuários com perfis variados e 50 refeições predefinidas ou geradas.

Relatórios

O sistema deve permitir a apresentação de relatórios no console como:

- Distribuição nutricional média dos cardápios sugeridos.
- Comparação entre o recomendado e o consumido (se implementado).
- Alimentos mais usados em sugestões para cada perfil.

Outros relatórios devem ser implementados pelo grupo.

2.3 TEMA 3: Sistema de Gestão Hospitalar com Lógica de Priorização de Pacientes

Desenvolvimento de um sistema de banco de dados relacional voltado para a gestão de atendimentos hospitalares, com foco na classificação e priorização de pacientes com base em critérios clínicos, tempo de espera e gravidade. O sistema deverá integrar modelagem avançada, consultas SQL complexas e lógica de programação externa para simulação de fluxos de atendimento.

2.3.1 Objetivo Geral

Modelar, implementar e testar um sistema de banco de dados hospitalar que gerencie triagens, prontuários, atendimentos e priorize automaticamente os pacientes segundo regras clínicas realistas. O sistema deve permitir o acompanhamento completo do paciente desde a entrada na unidade até a alta, promovendo a organização do fluxo de atendimento por meio de lógica relacional e programação externa.

2.3.2 Escopo e Requisitos

Modelagem

O sistema deverá contemplar pelo menos as seguintes entidades:

- *pacientes* (dados pessoais, histórico médico, convênio, comorbidades e qualquer outro dado importante para determinar o grau de prioridade do paciente.)
- *triagens* (sintomas, sinais vitais, horário de chegada, classificação de risco)
- *classificacoes_risco* (nome da cor da pulseira, prioridade, tempo máximo de espera)
- *atendimentos* (data/hora, profissional responsável, status – aguardando, em atendimento, finalizado)
- *profissionais_saude* (CRM, especialidade, plantão)
- *prontuarios* (diagnóstico, procedimentos, receitas, observações)
- *leitos* (tipo, disponibilidade, setor)
- *internacoes* (data de entrada/alta, paciente, leito)
- *historico_priorizacao* (registro da ordem de atendimento gerada)

Outros relacionamentos e tabelas auxiliares podem ser incluídos conforme a proposta do grupo.

Lógica de Priorização de Pacientes

O sistema deverá conter uma regra de triagem automatizada, inspirada no Protocolo de Manchester (ou outro similar), com base em:

- Classificação de risco (ex: vermelho, laranja, amarelo, verde, azul)
- Tempo de espera do paciente.
- Condições críticas (ex: taquicardia, febre alta, pressão arterial alterada, falta de ar)
- Comorbidades graves (diabetes, cardiopatia, etc.).

A priorização será calculada por uma função ou consulta SQL que retorne, em tempo real, a ordem ideal de atendimento.

Linguagem de Programação

Utilizar uma linguagem para:

- Cadastrar novos pacientes, triagens e atendimentos
- Atualizar o status dos pacientes em tempo real.
- Mostrar a fila de espera com base na lógica de priorização.
- Gerar relatórios e simular um plantão de 24h.

É preciso cadastrar pelo menos 50 pacientes com perfis diversos 100 triagens simuladas com variações de gravidade, 10 profissionais e 10 leitos.

Relatórios

O sistema deve permitir a apresentação de relatórios no console como:

- Tempo médio de espera por classificação de risco.
- Quantidade de pacientes atendidos por plantonista.
- Fila de espera atual com tempo estimado por paciente.
- Ocupação de leitos por setor.

Outros relatórios devem ser implementados pelo grupo.

2.4 TEMA 4: Simulador de Matrícula com Verificação de Conflito de Horários e Requisitos

Desenvolvimento de um sistema de banco de dados relacional para simulação de matrícula universitária, com validação automática de conflitos de horários entre disciplinas e verificação de pré-requisitos acadêmicos. O sistema deverá fornecer ao aluno um menu (via linha de comando) para seleção de disciplinas com base em seu histórico escolar, carga horária disponível e horários compatíveis.

2.4.1 Objetivo Geral

Criar um sistema que simule o processo de matrícula em disciplinas de um curso universitário, garantindo que o aluno só possa se inscrever em turmas cujos pré-requisitos tenham sido cumpridos e que não apresentem sobreposição de horários. O projeto deve contemplar modelagem relacional adequada, consultas SQL complexas e integração com lógica de programação para simulação do fluxo de matrícula.

2.4.2 Escopo e Requisitos

Modelagem

O sistema deverá contemplar pelo menos as seguintes entidades:

- *alunos* (nome, matrícula, curso, período atual e outras informações que julgarem necessárias.)
- *disciplinas* (nome, código, carga horária total e outras informações que julgarem necessárias)
- *pre_requisitos* (disciplina → pré-requisito obrigatório)
- *turmas* (disciplina, semestre, professor, turno, local, capacidade máxima e outras informações que julgarem necessárias.)
- *horarios* (turma, dia da semana, horário de início/fim e outras informações que julgarem necessárias.)
- *matriculas* (aluno, turma, status e outras informações que julgarem necessárias.)
- *historico_escolar* (aluno, disciplina, nota, frequência, situação: aprovado/reprovado e outras informações que julgarem necessárias.)

Outras entidades podem ser adicionadas conforme necessário (ex: docentes, semestres letivos, salas, cursos etc.).

Regras de Negócio Obrigatórias

- Pré-requisitos: O aluno só poderá se matricular em uma disciplina se todas as disciplinas listadas como pré-requisitos já tiverem sido cursadas e aprovadas.
- Conflitos de Horário: O aluno não poderá se matricular em turmas que tenham sobreposição de horários (mesmo dia e intervalo de tempo).
- Capacidade de turma: O sistema deve impedir matrícula em turmas que já estejam com capacidade máxima atingida.
- Carga horária máxima: Deve haver um limite de carga horária semanal por aluno, personalizável no sistema.

Linguagem de Programação

Utilizar uma linguagem para simular:

- Permitir que o aluno simule sua matrícula com base nas turmas ofertadas.
- Apresentar mensagens claras de erro quando:
  - Há conflito de horário;
  - Há disciplina com pré-requisito não cumprido;
  - Turma está lotada;
  - Carga horária extrapola o permitido.
- Mostrar disciplinas recomendadas com base no período atual do aluno e nas pendências do histórico escolar.
- Apresentar grade horária da matrícula atual simulada.

Relatórios

O sistema deve permitir a apresentação de relatórios no console como:

- Lista de alunos matriculados por turma.
- Grade horária individual de cada aluno.
- Disciplinas mais e menos procuradas.
- Disciplinas com maior índice de reprovação.
- Alunos aptos para TCC ou estágio (com base em carga horária ou disciplinas obrigatórias já cumpridas).

Dados simulados de históricos de disciplinas já cursadas serão necessários para elaboração de alguns relatórios.

3 Entregáveis

1. Relatório técnico (PDF) utilizando o template de artigos[1] da SBC (Sociedade Brasileira de Computação) com as seguintes seções:
   (a) Título do tema escolhido.
   (b) Membros do projeto e suas respectivas informações de curso, RGA e e-mail.
   (c) **Justificativa e contexto do problema**. Descreva o problema a ser solucionado e os principais desafios para solucioná-lo.
   (d) **Modelo ER**: Descreva o modelo entidade e relacionamento desenvolvido. Apresente detalhes de decisões de projeto tomadas nesta fase de construção do banco de dados.
   (e) **Regras de negócio implementadas e não-implementadas**: Deixe claro quais regras de negócio ou requisitos não foram implementados.
   (f) **Principais consultas SQL utilizadas**: principais consultas SQL utilizadas e que problemas elas solucionam.
   (g) **Screenshots das saídas da aplicação:** Apresente os principais relatórios exibidos por sua ferramenta.
   (h) **Conclusões e melhorias sugeridas**: Apresenta uma conclusão relatando se a sua solução resolve ou não o problema proposto. Além disso, proponha melhorias para o seu código e/ou banco de dados.
2. **Script SQL** de criação e povoamento do banco de dados.
3. **Código-fonte** da aplicação com documentação para execução local.
4. **Apresentação final (15 minutos)** demonstrando o sistema com as regras de negócios e/ou heurísticas inplementadas para solucionar o problema escolhido.

TODOS OS ARTEFATOS GERADOS NO ÂMBITO DESTE TRABALHO (CÓDIGO, DIAGRAMAS, VÍDEO, RELATÓRIO, ETC.) DEVEM SER ARMAZENADOS NO GOOGLE DRIVE INSTITUCIONAL DE UM DOS MEMBROS DO GRIPO E COMPARTILHADO COM O PROFESSOR (marcio.inacio@ufms.br). Apenas um membro do grupo deverá enviar um arquivo texto plano README.MD na área de submissão do AVA contendo o nome completo dos integrantes que de fato fizeram o trabalho, RGAs e o link para o google drive contendo todos os artefatos. A não observância a este item acarretará a nota zero para todos os membros do grupo. **Nenhum arquivo da pasta poderá ser alterado ou criado após o horário limite de entrega.**

[1] https://www.sbc.org.br/wp-content/uploads/2024/07/modelosparapublicaodeartigos.zip

4 A Apresentação

Cada grupo deverá gravar um vídeo de no máximo 15 minutos explicando como resolveu o problema proposto. Respondendo os seguintes itens na apresentação:

- Apresente-se falando o nome completo e qual curso cada integrante do grupo pertence;
- Apresentar as tabelas e seus respectivos atributos;
- Como realizaram a importação dos dados para o PostgreSQL?
- Fizeram algum script/programa para importar os dados?
- Qual linguagem escolheram? Por que esta linguagem?
- Mostre o script/programa fazendo a importação dos dados para o banco de dados vazio.
- Explique se os requisitos foram totalmente atendidos ou parcialmente. Se parcialmente, diga quais foram os itens atendidos.

A apresentação é um item indispensável para obter a nota do trabalho. Os alunos que não apresentarem terão sua nota final do trabalho composta apenas pela nota geral do grupo, obtida pela entrega do banco de dados e eventuais códigos.

O vídeo não pode ter cortes ou edições que interrompam o fluxo contínuo da apresentação. Softwares como Google Meet, Zoom.us, Skype podem ser úteis para a gravação. O Zoom gratuito permite que você grave a reunião localmente.

5 Considerações Finais

- **O banco de dados para o desenvolvimento deste trabalho deve ser OBRIGATORIAMENTE o PostgreSQL 12 ou superior.**
- **O MENOR INDÍCIO DE USO DE LLMs (ChatGPT, Claude AI., Gemini e similares) EM QUALQUER FASE DESTE TRABALHO, ACARRETARÁ ZERO PARA TODOS OS ENVOLVIDOS.**
- Não serão aceitos trabalhos atrasados. Se o grupo não entregar o trabalho no dia combinado, ele receberá nota zero.
- Em caso de projetos copiados de colegas todos os envolvidos recebem nota zero. Lembre-se é muito improvável que haja trabalhos totalmente iguais.
- O professor não ajudará os grupos na construção do trabalho.
- O professor poderá tirar dúvidas conceituais em horário de aula ou horário de atendimento.
- A nota dos integrantes não necessariamente será a mesma. Se durante a apresentação o professor detectar que algum integrante do grupo não tem domínio sobre o projeto, ele poderá receber uma nota menor que os demais integrantes.
- Entrevistas presenciais podem ser solicitadas a determinados grupos a fim de elucidar eventuais dúvidas/suspeitas do professor.

Tabela 1: Critérios de Avaliação do Projeto

Critério	**Peso**
Modelagem de dados e normalização	2,0
Resolução do problema proposto	2,5
Banco de Dados em SQL (com dados)	1,5
Funcionalidade e usabilidade do script ou aplicação.	1,5
Documentação técnica clara e bem estruturada	1,0
Apresentação final e domínio do conteúdo	1,5
Total (N)	**10,0**

6 Cálculo da Nota Final com Penalidade por Ausência na Apresentação

Seja:

- N: nota total obtida com base nos critérios avaliativos (valor entre 0 e 10);
- A: variável binária indicando a presença na apresentação final, definida como:

$$A = \begin{cases} 1, & \text{se o aluno participou da apresentação final} \\ 0, & \text{se o aluno não participou da apresentação final} \end{cases}$$

A nota final N_f será calculada da seguinte forma:

$$N_f = N \cdot (0{,}5 + 0{,}5 \cdot A)$$

Exemplos

- Se o aluno participou da apresentação ($A = 1$):

$$N_f = N \cdot (0{,}5 + 0{,}5 \cdot 1) = N \cdot 1 = N$$

- Se o aluno não participou da apresentação ($A = 0$):

$$N_f = N \cdot (0{,}5 + 0{,}5 \cdot 0) = N \cdot 0{,}5$$

Essa regra tem como objetivo incentivar a participação ativa na apresentação final do projeto, que é parte fundamental do processo avaliativo.